An introduction to Taverna workflows

1. Installing the Workbench

Exercise 1 Installing the Workbench

- Download Taverna from http://taverna.sourceforge.net
  - **Windows or linux**

If you are using either a modern version of Windows (Win2k or WinXP, with XP preferred) or any form of linux, solaris etc. you should download the workbench zip file. For windows users, Taverna can be unzipped and used, for linux you will also need to install GraphViz (http://www.graphviz.org/ the appropriate rpm for your platform)

  - **Mac OSX**

If you are using Mac OSX you should download the .dmg workbench file. Double-click to open the disk image and copy both components (Taverna and GraphViz) onto your hard-disk to run the application

- YOU WILL ALSO NEED a modern Java Runtime Environment (JRE) or Java Software Development Kit (SDK) from http://java.sun.com Java 5 or above

Workbench Layout

- AME – Advanced Model Explorer

  The Advanced Model Explorer (AME) is the primary editing component within Taverna. Through it you can load, save and edit any property of a workflow.

  - enables

    building

    loading

    editing

    saving workflows

Workflow Diagram Window

Visual representation of workflow

- Shows inputs / outputs, services and control flows
- Enables saving of workflow diagrams for publishing and sharing

Available Services Panel

Lists services available by default in Taverna

- ~ 3000 services
  - Local java services
  - Simple web services
  - Soaplab services – legacy command-line application
  - Gowlab services
  - BioMart database services
  - BioMoby services

Allows the user to add new services or workflows from the web or from file systems

Installing Plugins

- Go to the 'Tools' menu at the top of the workbench and select the *Plugin manager*
- Select *find new plugins*
- Tick the boxes for *Feta and LogBook* and install these plugins
- Two more options '*Discover*' and '*LogBook*' will now have appeared at the top of your screen
- Feta is now available through the Discover tab
- To use the LogBook, you also need a mySQL database

(we will come back to this later)

2. Adding new services

Exercise 2 Adding New Services

New services can be gathered from anywhere on the web

- Go to the DDBJ list of available web services at: http://xml.nig.ac.jp/wsdl/index.jsp

  These services were not designed for use in Taverna, but Taverna can use them if you supply the address of the WSDL file
- Click on the DDBJ blast service (http://xml.nig.ac.jp/wsdl/Blast.wsdl) and copy the web page address

Exercise 2 Adding New Services

- Go to the '*Available services*' panel and right-click on 'Available Processors'. For each type of service, you are given the option to add a new service, or set of services.
- Select '*Add new wsdl scavenger*'. A window will pop-up asking for a web address
- Enter the Blast Web service address
- Scroll down to the bottom of the '*Available Services*' panel and look at the new DDBJ service that is now included.

3. Finding and Invoking a Service

Exercise 3 - Finding and invoking a Service

Go to the 'Available Services' Panel

- Search for *Fasta* in the '*search list*' box at the top of the panel (we will start with simple sequence retrieval)
- You will see several services highlighted in red
- Scroll down to '*Get Protein FASTA*'

  This service returns a fasta sequence from a database if you supply it with a sequence id

Exercise 3
Invoking a single service

- Right click on the 'Get Protein FASTA' service and select '*Invoke service*'
- In the pop-up 'Run workflow' window add a protein sequence GI by selecting ID and right-clicking. Select 'new input value' and enter a value in the box on the right
  - GI is a genbank gene identifier (you don't need the gi: just the number, for example, the MAP kinase phosphatase sequence 'GI:1220173' would be entered as '1220173'
- Click '*Run workflow*' and the service is invoked

Exercise 3 View Results

- Click on '*Results*'
  - The fasta sequence is displayed on right when you select *click to view*
- Click on '*Process Report*'
  - Look at processes. This shows the experiment provenance – *where* and *when* processes were run
- Click on '*Status*'
  - Look at options As workflows run, you can monitor their progress here.

Exercise 3 - Conclusion

The processes for running and invoking a single service are the basics for any workflow and the tracking of processes and generation of results are the same however complicated a workflow becomes

In the next few exercises, we will look at some example workflows and build some of our own from scratch

4. Finding and Using Workflows

Exercise 4 – Finding and using workflows

- Select 'Open Workflow' from the File menu at the top of the workbench. You will see a selection of .xml files in an examples directory. These are workflow definition files
- Select '*ConvertedEMBOSSTutorial.xml*' and a pre-defined workflow will be loaded
- View the workflow diagram - you will see services of in different colours

Exercise 4 Workflow Documentation

- Find out what the workflow does by reading the workflow metadata
- In the AME – click on the name of the workflow – in this case '*A workflow version of the EMBOSS tutorial*' and then select the '*workflow metadata*' tab at the top of the AME. You will see a text description of the workflow, its author and its unique LSID. When publishing workflows for others, this annotation is useful information and allows the acknowledgement of intellectual property

Exercise 4 Workflow Features

- Run the workflow by selecting 'run workflow' from the file menu
- Watch the progress of the workflow in the '*enactor invocation*' window. As services complete, the enactor reports the events. If a service fails, the enactor reports this also

5&6 Building a simple workflow

5.1 Building a simple workflow from scratch

- Import the '*Get Protein FASTA*' service into a new workflow model. First, you will need to close the current workflow from the file menu, then find the '*Get Protein Fasta*' service again in the '*Available services*' panel.
- Right-click on 'Get Protein Fasta' and import it into the workbench by selecting '*Add to Model*'
- Go to the AME and expand the [+] next to the newly imported '*Get Protein Fasta*' service. You will see:
  - 1 input (Green arrow pointing up)
  - 1 output (purple arrow pointing down)

Exercise 5.2 Adding Input

- Define a new workflow input by right-clicking on '*Workflow Input*' and selecting 'create new Input'
- Supply a suitable name e.g. '*geneIdentifier*'
- Connect this new input to the '*Get Protein Fasta*' service by right-clicking on '*geneIdentifier*' and selecting '*getFasta ->id*'

You always build workflows with the flow of data

Exercise 5.3 Adding output

- Define a new workflow output by right-clicking on '*workflow output*' and selecting 'create new output'
- Supply a suitable name e.g. '*fastaSequence*'
- Connect this new output to the '*Get Protein Fasta*' service. remembering to build with the flow of data

  You have now built a simple workflow from scratch!
- Run the workflow by selecting 'run workflow' from the '*Tools and Workflow Invocation*' menu at the very top of the workbench. You will again need to supply a GI – for later exercises, please use a protein GI – e.g. 1220173

Exercise 6 Stringing Services Together

- We have used '*Get Protein Fasta*' to retrieve a sequence from the genbank database. What can we do with a sequence?
  - Blast it?
  - Find features and annotate it?
  - Find GO annotations?

Blast it?

The first thing you need to do is find a service which performs a blast. For this, we are going to use the ***Feta Semantic Discovery*** Tool

Feta is a tool to semantically describe services. Instead of the user needing to know exactly what a service provider has called their services, the user can search by the biological tasks that are performed by the services, or by properties of the service, for example, the types of inputs it requires/outputs it produces

Finding Blast

- Select the '*Discover*' tab and select 'uses method from the first drop down menu
- When you select it, 'bioinformatics algorithm' will appear in the adjoining box. Scroll down this list to find 'Similarity search algorithm, and then the subclass of this, BLAST (basic_local_alignment_search_tool)
- Select BLAST and click '*Find Service*'

  The results are all the annotated services that perform blast analyses (there may be more un-annotated ones!)

Finding Blast

- Select '*searchSimple*' from the list and look at the details
- Look at the service description

  This tells you what the service does and what each input/output is expecting/produces. It also tells you where the service comes from. For this example, we are using BLAST from the DNA Databank in Japan
- Right-click on '*searchSimple*' in the Feta results list and select 'add to model'

  This adds the service to your current workflow in the 'Design Window'
- Before you go back to the Design window, go back to search services and experiment with other ways of finding services – e.g. by task, input/output, resource etc

Exercise 6 Blast It

- Go back to the Design window. *SearchSimple* will have been imported into your model
- In the AME expand the [+] for the '*search simple*' service and view the input/output parameters
- This time, you will see three inputs and two outputs. For the workflow to run, each input must be defined. If there are multiple outputs, a workflow will usually run if at least one output is defined.

Exercise 6 Blast it

- Create an output called '*blast_report*' in the same way we did before
- The sequence input for the Blast will be the output from the '*Get Protein Fasta*' service. Connect the two together, from '*Get Protein Fasta Output Text*' to '*search simple query*'
- Create two more inputs called '*database*' and '*program*' and connect them to the '*database*' and '*program*' inputs on '*search simple*' service

Exercise 6 Blast it

- Once more select '*run workflow*' from the '*Tools and Workflow Invocation*' menu. You will see a run workflow window asking for 3 input values
- Insert a GI (e.g. 1220173), a program (blastp for protein-protein blast), and a database, e.g. SWISS (for swissprot)
- Click 'run workflow'. This time you will see a blast report and a fasta sequence as a result

Exercise 6 Blast it

- For parameters that do not change often, you will not wish to always type them in as input. In this example, the database and blast program may only change occasionally, so there is an alternative way of defining them.
- Go back to the AME and remove the '*database*' and '*program*' inputs by right-clicking and selecting '*remove from model*'

Exercise 6 String Constants

- Select '*string constant*' *from* '*Available Services*'
- Right-click and select 'add to model with name…'
- Insert '*program*' in the pop-up window
- Select '*string constant*' for a second time and repeat for a string constant named '*database*'
- In the AME, right-click on '*program*' and select '*edit me*'
- Edit the text to 'blastp'. Repeat for 'database' and enter 'SWISS' for the swissprot database
- Run the workflow – it runs in the same way
- Save the workflow by selecting the 'save' icon at the top of the AME.

7 A protein annotation workflow

Exercise 7 Protein Annotation

How can we use Taverna to annotate our protein with function descriptions?

- In the 'available services' panel, find the emboss soaplab services and find the '*protein_motifs*' section

  Hint: use the simple text search at the top of the panel

- Find out which of these services enable searching of the Prosite and Prints databases by fetching the service descriptions. To do this *right-click* on '*protein_motifs*' and select '*fetch descriptions*'
- Import both services into the workflow model.

Exercise 7 Protein Annotation

- Connect these services up to the workflow so that you can find prints and prosite matches in the query sequence returned from '*Get Protein Fasta*' – you will see that soaplab services have many input values
- Soaplab services have many input parameters, but many have default values so may not always need to be altered. In this case, you can run the services by simply adding the query sequence. Go to the EMBOSS home page to find out which input(s) relate to the query sequence.
- This extra searching is impractical – but is necessary if it hasn't been described in Feta.
- Soaplab has an extra metadata section however, right click on the service in the AME and select 'get soaplab metadata'

Exercise 7 Protein Annotation

- Run the workflow – now you have blast results and protein domain/motif matches

- How else can you annotate your protein? As an advanced exercise, you might want to search for other ways of characterising your sequence e.g. structural elements, GO annotation?

Saving Results

Taverna provides several options for saving data.

1. Individual data items can be saved by right-clicking on them
2. All data can be saved to disk
3. Textual/tabular data can be saved to excel

- **Save all the data from your workflow**

Advanced Exercises

The previous exercises have covered the basics of myGrid workflows. The following demos and exercises cover more advanced features, such as rendering output, configuring BioMart services, dealing with service failure and iterating over datasets. You may not reach the end of these exercises, but they will provide a some examples to take home

Exercise 8 Defining Output Formats

So far, most of the outputs we have seen have been text, but in bioinformatics, we often want to view a graph, a 3D structure, an alignment etc. Taverna is able to display results using a specific type of renderer if the workflow output is configured correctly.

- Reset the workbench and load '*convertedEMBOSSTutorial*' from the '*examples*' directory
- Look at the workflow diagram and read the workflow metadata to find out what the workflow does
- Run the workflow

Exercise 8 Defining Output Format

- **Look at the results.** For '*tmapPlot*' and '*outputPlot*', you will see the results are displayed graphically. This is achieved by specifying a particular mime type in the output.
- Go back to the AME and look at the metadata for '*tmapPlot*' and '*outputPlot*'.
- **Select MIME Types.** As you can see, each has the image/png mime type associated with it. If you wish to render results in anything other than plain text, you MUST specify the mime-type in the workflow output

Exercise 8 Taverna MIME-Types

The following mime-types are currently used by Taverna

text/plain=Plain Text

text/xml=XML Text

text/html=HTML Text

text/rtf=Rich Text Format

text/x-graphviz=Graphviz Dot File

image/png=PNG Image

image/jpeg=JPEG Image

image/gif=GIF Image

application/zip=Zip File

chemical/x-swissprot=SWISSPROT Flat File

chemical/x-embl-dl-nucleotide=EMBL Flat File

chemical/x-ppd=PPD File

chemical/seq-aa-genpept=Genpept Protein

chemical/seq-na-genbank=Genbank Nucleotide

chemical/x-pdb=Protein Data Bank Flat File

chemical/x-mdl-molfile

Exercise 8 Taverna MIME types(2)

The 'chemical/' mime-types are rendered using SeqVista to view formatted sequence data

- Reset the workbench and load 'FetchPDBFlatFile' from the '*examples/library*' directory for a demo

  The chemical/x-pdb can be used to view rotating 3D protein images

Advanced Features

- Spotlight on BioMart
- BioMoby Services
- Iteration
- Control Flow
- Substituting Services and fault tolerance

Spotlight on Biomart

Biomart enables the retrieval of large amounts of genomic data e.g. from Ensembl and sanger, as well as Uniprot and MSD datasets

- After saving any workflows you want to keep, reset the workbench in the AME
- Load the workflow 'BiomartAndEMBOSSAnalysis.xml' from the '*examples*' directory
- Run the Workflow

Spotlight on Biomart

This Workflow Starts by fetching all gene IDs from Ensembl corresponding to human genes on chromosome 22 implicated in known diseases and with homologous genes in rat and mouse.

For each of these gene IDs it fetches the 200bp after the five prime end of the genomic sequence in each organism and performs a multiple alignment of the sequences using the EMBOSS tool 'emma' (a wrapper around ClustalW). It then returns PNG images of the multiple alignment along with three columns containing the human, rat and mouse gene IDs used in each case.

Configuring Biomart

- Right-click on the 'hsapiens_gene_ensembl' service and select 'configure BioMart query'
- By selecting '*filters*' – change the chromosome from 22 to 21 – now the workflow will retrieve all disease genes from chromosome 21 with rat and mouse homologues
- Run the workflow and look at the results

- See how the 'with *disease association*' filter was configured and the '*sequence exports*' were configured on the other Biomart queries for mouse and rat

Adding Extra Information

Find out which diseases are on your chosen chromosome by adding a new Biomart query processor

- Select '*hsapiens_gene_ensembl*' from the available services panel and select 'invoke with name….' (as there is already a service with that name!)
- Call the service '*hsapiens_disease*'
- Select Filters
- Configure '*hsapiens_disease*' by selecting 'id list limit' and '*ensembl gene IDs*' filter under the '*gene*' tab
- Configure the output (by selecting attribute) and select '*Mim morbid accession*' under the 'External Reference' tab in the attributes section

Adding Extra Information

- Connect the input to the '*hsapiens_gene_ensembl*' service via the '*ensembl_gene_id*'
- Create a new workflow output for the '*disease_description*' output
- Re-run the workflow and view which diseases are associated with your chromosome

Spotlight on BioMoby

The process of adding a BioMoby service is different from other services. BioMoby services need to be defined using terms from the Moby Object ontology

- Reset the workflow and load the '*blast-biomoby.xml*' workflow from

  http://www.cs.man.ac.uk/~katy/taverna/

Spotlight on BioMoby

- Run the workflow and look at the results

  As the workflow name suggests, a blast search is performed on a sequence
- Look at the workflow diagram

  Instead of simply giving the blast service a fasta sequence, there is a '*Fasta*' sequence object defined.
- Look at the inputs for 'Fasta'
- Read the metadata for the '*Fasta*' object in the AME window

Spotlight on BioMoby

The Fasta object is defined by

1. The sequence (as a plain string)
2. The namespace (i.e. the database the sequence came from)
3. A unique identifier for the sequence
4. A name

These extra definitions take time for the user to define, but they have other advantages

Spotlight on BioMoby

- Right-click on the '*Fasta*' object in the AME and select '*Moby Object Details*'
- A pop-up window will show you what BioMoby services a 'Fasta' sequence is produced by and what services it can feed into
- Right-click on the 'getDragonBlastText' service and select 'Moby Object Details'. This tells you what the service requires as inputs and what it produces as output

Spotlight on BioMoby

- The BioMoby services are annotated using terms from the Moby ontology to enable semantic searching for services.
- BioMoby services are specialist kinds of service from a closed community. The object model, ontology and annotations have been agreed by the BioMoby service providers.
- Semantic discovery queries over other myGrid services are also possible using the myGrid ontology and the Feta Semantic discovery component.
- The myGrid ontology and the Biomoby ontology both share the same service ontology, so feta can search both types of service

Iteration

Taverna has an implicit iteration framework. If you connect a set of data objects (for example, a set of fasta sequences) to a process that expects a single data item at a time, the process will iterate over each sequence

- Reload the BiomartandEMBOSSAnalysis.xml workflow from the examples directory
- Watch the progress report. You will see several services with 'Invoking with Iteration'

Iteration

The user can also specify more complex iteration strategies using the service metadata tag

- Reset the workflow and load the '*IterationStrategyExample.xml*'
- Read the workflow metadata to find out what the workflow does
- Select the '*ColourAnimals*' service and read the metadata for that service. Under the description is the iteration strategy
- Click on '*dot product*'. This allows you to switch to cross product

Iteration

- Run the workflow twice – once with '*dot product*' and once with '*cross product*'.
- Save the first results so you can compare them – what is the difference? What does it mean to specify dot or cross product?

Substituting services and fault Tolerance

Taverna does not own many of the bioinformatics services it provides. This means that it cannot control their reliability. Instead, Taverna provides strategies for dealing with services being unavailable

- Reload the '*convertedEMBOSSTutorial.xml*' from the '*examples*' directory.
- Look at the metadata for the '*emma*' service. It is an implementation of clustalw
- Find the DDBJ clustalw service – HINT: use the Feta discovery tool

Substituting Services

- When you have added this service to your workflow, right-click on it and select '*add as alternate*'
- In the resulting menu select '*emma*'
- The DDBJ version of the clustalw service is now added as an alternative to emma in the AME. It will be called '*alternate1*'
- Select '*alternate1*' and look at the inputs and outputs. These need to be mapped to the correct inputs and outputs in emma

Substituting Services

- Right-click on the '*query*' input in alternate1 and map it to '*sequence_direct_data*'. In both services, these inputs expect a set of fasta sequences.
- Right-click on the '*result*' output and map it to '*outseq*' in emma in the same way.
- Now you have a workflow which will run using emma when it is available – but will substitute it for DDBJ clustalw if emma fails!

Fault Tolerance

Taverna also allows the user to specify the number of times a service is retried before it is considered to have failed. Sometimes network traffic is heavy, so a working service needs to be retried

- Select 'tmap' from the same workflow. To the right of the service name are a series of 0s and 1s. By simply typing the numbers, the user can specify the number of retries and the time between the retries
- Change it to 3 retries for 'tmap' and set the status to 'critical' using the final tickbox. Now it is critical, it means the whole workflow will be aborted if '*tmap*' fails after 3 retries. Failures in non-critical services will not abort the workflow run.

Shim Services

This exercise highlights the services that do not perform biological functions, but are vital for running life science workflows

Finding Genes

- Load the workflow entitled genscan_shim_example.xml from the page http://www.cs.man.ac.uk/~katy/taverna
- Look at the workflow metadata – what does the workflow do?
- Run the workflow.
- For an input file, load *example_input.txt* from the same web page

  What happens?

  Did all the services return results?

  Why did some fail?

Finding genes

- Load the workflow entitled genscan_shim_example2.xml from the page http://www.cs.man.ac.uk/~katy/taverna
- Look at the workflow metadata – what does the workflow do? How is it different from the previous one?
- Run the workflow (using the same input) – what happens this time?

Genscansplitter is a shim service – it performs no biological function, it simply parses a results file.

Other shims

- There are many myGrid shim services. These are currently being described in a shim library, but for now, a small collection are documented here

  http://www.cs.man.ac.uk/~hulld/shims.html

  From the list,

- Find a shim that will return a genbank DNA file from an id. Load the example workflow and run it in Taverna
- Find a shim that will translate DNA

  HINT: these services might be in the feta registry

Other Shims

- Load the CompareXandYFunctions.xml workflow from the examples directory

  This workflow contains several shims. Some are beanshell scripts
- Select the 'GetUniqueIDs' service in the AME and right-click
- Look a the script and see if you can work out what it is doing

Beanshell scripts allow users to write small, bespoke java scripts to allow incompatible service to work together

Other Shims

- The emboss suite of programs have a subdivision – edit
- All the edit services are shims
- Experiment with the edit services
- Find a service that will remove gaps from sequences

Useful links

Taverna user manual:

http://www.mygrid.org.uk/usermanual1.7/

- Taverna mailing lists:

http://taverna.sourceforge.net/index.php?doc=lists.html